31장 RegExp

정규 표현식(Regular expression)은 일정한 패턴을 가진 문자열의 집합을 표현하기 위해 사용하는 형식 언어(formal language)

정규 표현식은 문자열을 대상으로 **패턴 매칭 기능**을 제공
패턴 매칭 기능이란 특정 패턴과 일치하는 문자열을 검색하거나 추출 또는 치환할 수 있는 기능

정규 표현식의 생성

정규 표현식 리터럴

- 패턴과 플래그로 구성

```
const target = "Is this all there is?';

// 패턴: is
// 플래그: i => 대소문자를 구별하지 않고 검색한다.
const regexp = /is/i;

// test 메서드는 target 문자열에 대해 정규 표현식 regexp의 패턴을 검색하여 매칭 결과를 불리언 값으로 반환한다.
regexp.test(target); // true
```

RegExp 메서드

RegExp.prototype.exec

- 인수로 전달받은 문자열에 대해 정규 표현식의 패턴을 검색하여 매칭 결과를 배열로 반환한다. 매칭 결과가 없는 경우 null을 반환

```
const target = 'Is this all there is?';
const regExp = /is/;

regExp.exec(target);
```

```
// ["is", index: 5, input: "Is this all there is?", groups:
undefined]
```

RegExp.prototype.test

- 인수로 전달받은 문자열에 대해 정규 표현식의 패턴을 검색하여 매칭 결과를 불리언 값 반환

```
const target = 'Is this all there is?';
const regExp = /is/;

regExp.test(target); // true
```

String.prototype.match

- 대상 문자열과 인수로 전달받은 정규 표현식과의 매칭 결과를 배열로 반환

```
const target = 'Is this all there is?';
const regExp = /is/;

target.match(regExp);
// ["is", index: 5, input: "Is this all there is?", groups:
undefined]
```

플래그

플래그	의미	설명
i	Ignore case	대소문자를 구별하지 않고 패턴을 검색한다.
g	Global	대상 문자열 내에서 패턴과 일치하는 모든 문자열을 전역 검색한다.
m	Multi line	문자열의 행이 바뀌더라도 패턴 검색을 계속한다.

어떠한 플래그도 사용하지 않은 경우 대소문자를 구별해서 패턴을 검색

패턴

문자열 검색

정규 표현식의 패턴에 문자 또는 문자열을 지정하면 검색 대상 문자열에서 패턴으로 지정한 문자 또는 문자열을 검색

```
const target = 'Is this all there is?';

// 'is' 문자열과 매치하는 패턴. 플래그가 생략되었으므로 대소문자를 구별한다.
const regExp = /is/;

// target과 정규 표현식이 매치하는지 테스트한다.
regExp.test(target); // true

// target과 정규 표현식의 매칭 결과를 구한다.
target.match(regExp);
// ["is", index: 5, input: "Is this all there is?", groups: undefined]
```

임의의 문자열 검색

.은 임의의 문자 한 개를 의미

```
const target = 'Is this all there is?';

// 임의의 3자리 문자열을 대소문자를 구별하여 전역 검색한다.
const regExp = /.../g;

target.match(regExp); // ["Is ", "this", "s a", "ll ", "the", "re ", "is?"]
```

반복 검색

{m,n}은 앞선 패턴(다음 예제의 경우 A)이 최소 m번, 최대 n번 반복되는 문자열을 의미

```
const target = 'A AA B BB Aa Bb AAA';

// 'A'가 최소 1번, 최대 2번 반복되는 문자열을 전역 검색한다.
```

```
const regExp = /A{1,2}/g;

target.match(regExp); // ["A", "AA", "A", "AA", "A"]
```

OR 검색

|는 or의 의미

```
const target = 'A AA B BB Aa Bb';

// 'A' 또는 'B'를 전역 검색한다.
const regExp = /A|B/g;

target.match(regExp); // ["A", "A", "A", "B", "B", "B",
"A", "B"]
```